Dual Serial Board

Technical Support Manual

ASDG Incorporated
925 Stewart Street
Madison, WI 53713

May 1, 1989

1 NOTICES

1.1 Disclaimer

THIS MANUAL HAS BEEN CHECKED AND IS BELIEVED TO BE ACCURATE AND CORRECT AND IS SUBJECT TO CHANGE WITHOUT NOTICE. THE INFORMATION IN THIS MANUAL IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED. THE ENTIRE RISK AS TO THE ACCURACY OF THE INFORMATION HEREIN IS ASSUMED BY THE USER. ASDG INCORPORATED DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS REGARDING THE USE OF, OR THE RESULTS OF THE USE OF, THE INFORMATION IN TERMS OF CORRECTNESS, ACCURACY, RELIABILITY, OR OTHERWISE. ASDG INCORPORATED WILL NOT BE LIABLE FOR DIRECT, INDIRECT, INCIDENTAL OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT IN THE INFORMATION EVEN IF ASDG INCORPORATED HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF IMPLIED WARRANTIES OR LIABILITY FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THE ABOVE LIMITATION OR EXCLUSION MAY NOT APPLY TO YOU.

ASDG INCORPORATED SPECIFICALLY DISCLAIMS ALL LIABILITY FOR DAMAGES RESULTING FROM THE USE OR MISUSE OF THE PUBLIC DOMAIN AND/OR SHAREWARE PROGRAMS DISTRIBUTED WITH THIS PRODUCT. SUCH PROGRAMS ARE PROVIDED AS A SERVICE AND ARE UNSUPPORTED BY ASDG INCORPORATED.

1.2 Copyright

This manual is COPYRIGHT 1989 by ASDG Incorporated. All rights are reserved worldwide.

1.3 Restrictions

The contents may not, in whole or in part, be copied, photocopied, reproduced, or reduced to any electronic medium or machine readable form, stored in any retreival system, or translated into any human or machine language without prior consent in writing from ASDG Incorporated.

1.4 Attributions

Amiga and WorkBench are trademarks of Commodore-Amiga, Inc. IBM PC-AT is a trademark of International Business Machines, Inc. V1.3 Commodore Serial Device used under license.

1.5 FCC Compliance

WARNING: This equipment generates, uses, and can radiate radio frequency energy and if not installed and used in accordance with the instructions manual, may cause interference to radio communications. It has been tested and found to comply with the limits for a Class A computing device pursuant to Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference when operated in a commercial environment. Operation of this equipment in a residential area is likely to cause interference in which case the user at his own expense will be required to take whatever measures may be required to correct the interference.

If you suspect interference, you can test the product by turning the Amiga off and on with this product installed and not installed. If this product does cause interference, try the following:

1. Reorient the antenna or AC plug on the affected equipment.
2. Change the relative positions of the Amiga and the affected equipment.
3. Move the Amiga farther away from the affected equipment.
4. Plug either the Amiga or the affected equipment into a different outlet so that the Amiga and the affected equipment are on different circuits.

Use only shield-grounded cables when connecting peripherals to the Amiga.

All peripherals must be labeled to comply with the FCC emissions requirements. Class B certified devices will usually have lower emissions than Class A devices.

Operation with unlabeled peripherals is likely to result in interference.

Use this equipment only with three-pronged type (AC ground) AC wall recepticals.

If necessary, consult your dealer or an experienced Radio Frequency Interference technician for additional suggestions. You may find the FCC booklet "How to Identify and Resolve Radio-TV Interference Problems" helpful. It is available from the U.S. Government Printing Office, Washington, D.C. 20402, stock no. 004-000-00345-4.

2 SPECIFICATIONS

2.1 Electrical Specifications

Power requirements	+4.75 to 5.25 Vdc at 1.37 A. Max.
	+11.4 to 12.6 Vdc at 50 mA. Max.
	–12.6 to 11.4 Vdc at 50 mA. Max.

2.2 Environmental Specifications

The Dual Serial Board is rated for the following environmental conditions.

- Storage Temperature........-40 to +70 degrees C
- Operating Temperature.......0 to +55 degrees C
- Relative Humidity...........5 to 85% (non-condensing)

2.3 Mechanical Specifications

Physical dimensions:

13.85" long x 4.45" high x 0.85" wide (including rear panel bracket)

Connector Pin Number	Signal Name And Abbreviation	Signal Direction
1	Data Carrier Detect (DCD)	Input
2	Receive Data (RxD)	Input
3	Transmit Data (TxD)	Output
4	Data Terminal Ready (DTR)	Output
5	Signal Ground (GND)	—
6	Data Set Ready (DSR)	Input
7	Request To Send (RTS)	Output
8	Clear To Send (CTS)	Input
9	Ring Indicator (RI)	Input

Table 1: Dual Serial Board Connector Pinouts

2.4 Connector Pinouts

The connectors for both port 0 and port 1 are male 9 pin sub-miniature D style connectors with IBM PC-AT standard configuration. The signal name and direction for each pin are shown in Table 1.

3 Installing the Dual Serial Board and Software

3.1 Board Installation

A small to medium size Phillips screw driver is required to install a Dual Serial Board in an Amiga 2000.

Remove the screws along the bottom of the sides of the Amiga 2000. There are two screws along the lower edge of each side of the machine.

Remove the screw in the top center of the back of the Amiga 2000. This screw is located between a similar looking screw which holds in the top corner of the power supply and a smaller screw which holds in place the bracket for the coprocessor slot.

Grasp the cover on both sides and slide it toward the front of the Amiga 2000 and up.

Unscrew the bracket for the slot the Dual Serial Board will be mounted in and remove it from the system. Install the Dual Serial Board in the slot you have decided on, making sure that the board seats firmly in the connector. Re-use the screw you just removed to screw the Dual Serial Board bracket in place.

Re-install the cover of the Amiga by reversing the steps you used to remove it.

3.2 Software Installation

The Dual Serial Board software driver is installed by copying siosbx.device to your `Devs:` directory. This can be accomplished by double clicking on the `Install` icon in the root directory of the distribution diskette.

The `Install` program will also copy the DOS Handlers (which allow CLI users to access the expansion serial ports) into your `L:` directory.

The `Install` program will also install ASDG's Serial DisPatcher if you so choose. The Serial DisPatcher is described more fully in a later section which you should read before running the `Install` program.

3.3 Confirming Board Installation

Double clicking on the `ICheck` icon in the root directory of the distribution diskette will confirm that the DSB has properly autoconfigured and will be found by the software drivers.

9 Pin Connector Pin Number	Signal Name And Abbreviation	Amiga Connector Pin Number
1	Data Carrier Detect (DCD)	-
2	Receive Data (RxD)	2
3	Transmit Data (TxD)	3
4	Data Terminal Ready (DTR)	6
5	Signal Ground (GND)	7
6	Data Set Ready (DSR)	20
7	Request To Send (RTS)	5
8	Clear To Send (CTS)	4
9	Ring Indicator (RI)	-

Table 2: Amiga To Dual Serial Board Cable Pinout

3.4 Connecting To An Amiga Serial Port

Given in table 2 above, is the construction of a cable which will connect an Amiga standard serial port to an expansion port on a DSB.

4 Software Support Levels

The Amiga personal computer comes with a single RS-232 style serial interface. The interface is supported by operating system software on two levels. These are "the Exec level" and "the DOS level."

The Exec level software support consists of the Exec device driver, "serial.device," and is the programmer's principal interface to the serial port.

The DOS level software support is provided by the "port-handler" and is called "SER:". This is the CLI user's principal interface to the serial port.

The DSB provides both the Exec and DOS levels of support for up to 32 additional serial ports. Exec level support is provided by the program **`siosbx.device`** which should reside in your **`Devs:`** directory. DOS level support is provided by **`SERX-Handler`** (and **`SERXI-Handler`**) which should reside in your **`L:`** directory.

4.1 Exec Level Support

Programming the **`siosbx.device`** is performed in exactly the same way as programming the Commodore standard **`serial.device`**. Only two minor differences (both in the call to **`OpenDevice`**)are to be noted.

First, some background. **`OpenDevice`** takes the following form:

```
error = OpenDevice(DevName , UnitNum , IOReq , Flags)
```

Where:

`DevName`	Is a pointer to a string containing the name of the device.
`UnitNum`	Is an integer containing the unit number to be opened.
`IOReq`	Is a pointer to an **`IORequest`** block which will be filled in the device driver and used to communicate with the device.
`Flags`	Contains additional driver specific information. Usually zero.

With this in mind, a call to **`OpenDevice`** for the Amiga serial device usually looks like this:

```
if (OpenDevice("serial.device", 0, IOB, 0)) ...
```

The two differences are:

1. Rather than passing a pointer to a string constant containing `"serial.device"`, make this argument a string pointer variable which can point to either **`serial.device`** or **`siosbx.device`**. You might consider giving the user the opportunity to choose between them.
2. Instead of passing the constant `0` as the unit number make this argument an integer variable. This argument will be used to select between the possibly many (**`siosbx.device`**) ports you might have on your system.

So, some example code might look like:

```
char *AmigaSerial = "serial.device";
char *ASDGSerial  = "siosbx.device";

DevName = AskUserWhichDevice() ? ASDGSerial : AmigaSerial;
if (DevName == ASDGSerial) UnitNum = AskUserWhichUnit();
else UnitNum = 0;
Result = OpenDevice(DevName , UnitNum , IOB , 0);
```

Where **`AskUserWhichDevice`** and **`AskUserWhichUnit`** are hypothetical routines which allow the user to specify which device and unit, respectively.

Other than the **`OpenDevice`** call, no other changes are required. You may use the Amiga Rom Kernel Manual for your programming documentation.

4.2 Additional Features, Notes And Limitations

Documented here are some of the additional features which are provided by the DSB's Exec Level driver as well as its one limitation.

4.2.1 Modifying Control Line Status

Several users have requested the ability to directly manipulate the status of two control lines: RTS and DTR. Normally, the DTR line can be lowered by closing the device. When the device is reopened, the DTR line will again be raised. The Amiga serial device provides no way to manipulate the RTS line.

To facilitate the manipulation of these lines while the device is still open we provide the following non-standard command:

Name SetCtrlLines – Allows setting of various control lines.

Function This function allows the state of various control lines to be affected without closing the expansion serial device. The effect of the command will take place immediately.

IO Request Note: Actually defining the mnemonics are left to the programmer. Their suggested names are supplied here with their intended values.

io_Command **`SIOCMD_SETCTRLLINES`** (16 decimal or 0x10 in hexadecimal).

io_Offset A bit mask containing a one in each bit position which will be changed. Any bit positions with a zero value will be unchanged irrespective of the value contained in **`io_Length`**.

io_Length The values each affected bit should be set to. Any bit position which has a zero in the corresponding position in **`io_Actual`** will be ignored.

The valid bit positions (masks) are:

`SIOB_SET_RTS` Bit position 0.

`SIOB_SET_DTR` Bit position 1.

Thus to turn off RTS you would perform:

```
IOB->io_Command = 0x10;/* or 16 or define own mnemonic */
IOB->io_Offset  = 2;    /* specify bit 1 be changed */
IOB->io_Length  = 0;    /* it will be changed to a 0 */
DoIO(IOB);
```

Or, to turn RTS off and DTR on at the same time you would perform:

```
IOB->io_Command = 0x10;/* or 16 or define own mnemonic */
IOB->io_Offset  = 3;    /* specify bits 0 & 1 be changed */
IOB->io_Length  = 1;    /* bit 0 set, bit 1 unset */
DoIO(IOB);
```

4.2.2 Error Reporting

The current Amiga Exec Level support (embodied in serial.device) removed documented support for various error status messages such as detecting invalid baud rates. The DSB Exec Level driver reports *all* error messages correctly as documented in the Rom Kernel Manual (even if the Amiga serial device does not).

The programmer can choose to deal with these error conditions as desired. Supporting these conditions will provide more robust functionality when operating on a DSB port and make no difference when using the Amiga's serial port.

4.2.3 Known Limitations

The DSB driver cannot provide mark or space parity. Setting mark or space parity through the DSB will set no parity checking. Odd and even parity function properly.

Due to the nature of the Amiga serial hardware implemenation, the Amiga's own serial port will drift away from the specified baud rate settings above 38,400 baud. This affects those who wish to connect a DSB port to an Amiga port for operation above 38,400 baud. Connections between DSB ports will operate correctly throughout the allowable baud rate range.

If you absolutely have to connect a DSB port to an Amiga port *and* run this connection at speeds higher than 38,400 baud, you can empirically find a baud rate combination which produces the correct results. Note, however, that it is unlikely that the Amiga's own serial port will reliably keep up with these speeds.

4.3 DOS Level Support

The DSB comes with two DOS handlers, **SERX-Handler** and **SERXI-Handler**. The former is similar to the standard DOS handler **Ser:**, while the latter is similar to the standard DOS handler **AUX:**. That is, SERX-Handler provides buffered input/output while SERXI-Handler provides unbuffered (or interactive) input/output.

The DSB DOS handlers provide considerably greater flexibility than their standard DOS counterparts in that they provide the ability to dynamically change serial parameters at will.

This accomplished via three stages of control.

1. Serial parameters can be specified on the command line allowing the CLI user to have total control over serial line characteristics. This is accomplished by specifying the serial parameters as if they were directory and file names on the serial device.

2. Any paramters not specified on the command line can be set to defaults specified in the `devs:mountlist` file.
3. Any parameters not specified either on the command line or in the `devs:mountlist` file will default to the values specified in **`Preferences`** for the Amiga's standard serial port.

That is, serial parameters can be set on a command by command basis or via two levels of default settings.

4.3.1 Setting Serial Parameters On The Command Line

The syntax of command line specification of serial parameters is given as follows:

```
<device name>:<baud>/<data format>/<buffer size>/<handshaking>
```

where:

`<device name>` Is any name you desire. This must correspond to an entry in your `devs:mountlist` file. **Note:** Unit differentiation is done in the `devs:mountlist` file and, consequently, is reflected in the device's name.

`<baud>` Is any number between 110 and 292000. This will be used as the baud rate of the serial line. If this argument is missing, the baud rate will be chosen from the handler's two levels of defaults.

`<data format>` Is a 3 character string of the form:

```
<nbits><parity><nstop>
```

Where:

- `<nbits>` Is the number of bits which will be sent in each byte. This number may be between 5 and 8.
- `<parity>` Is the parity setting. This may be the letters "N" for none, "E" for even, or "O" for odd. This is **not** case sensitive.
- `<nstop>` This is the number of stop bits which will be sent after each byte. This may be 1 or 2. In **`<nbits>`** is 8, then **`<nstop>`** must be 1.

If this parameter is missing, data length, parity, and number of stop bits will be chosen from the handler's two level's of defaults.

`<buffer size>` Is any number larger than 64. This represents the number of bytes which will be requested as the size of the serial receive buffer. Obviously, this number is limited by your memory availability. One reasonable rule of thumb is "the higher the baud rate, the larger the buffer". If this parameter is missing a value will be chosen from the handler's two levels of defaults.

`<handshaking>` Is a number between 0 and 3 (inclusively) which specify what type of handshaking is to be used. If missing, the handler chooses handshaking from its two levels of defaults. The values represent:

0 No handshaking.
1 XON/XOFF handshaking.
2 CTS/RTS handshaking.
3 Both XON/XOFF and CTS/RTS handshaking.

Notice that any missing argument will cause the handler to choose a default setting either from the `devs:mountlist` file or from `Prefereces`.

For example, the following command line will copy the named file to an expansion serial line set entirely to default settings:

```
copy S:Startup-Sequence Ser3:
```

The following command line will copy the named file to an expansion serial line set to 9600 baud with a 20000 byte buffer. All other parameters will be set to the defaults:

```
copy S:Startup-Sequence Line2:9600//20000
```

The following command line will copy the named files to an expansion serial line set to 31250 baud with 7 data bits, no parity, 2 stop bits, a large buffer and XON/XOFF handshaking.

```
copy S:Startup-Sequence Micky:31250/7N2/30000/1
```

In the above examples, note that the expansion serial port name is entirely dependent upon a corresponding `devs:mountlist` entry.

4.3.2 Setting Mountlist Entry Defaults

As described above, any serial parameters not specified on the command line will be set according to two levels of defaults. The first level of default is any settings you may have made in the `devs:mountlist` file. These settings are communicated to the handler by setting an appropriate value on a `devs:mountlist` entry `Startup` line. This takes the form of:

```
Startup = 0xNNNNNNNN
```

The eight digit hexadecimal number is composed of fields which will be described below. Note that any fields specified to be 0 mean that that parameter will be set according to the handler's second means of default, `Preferences`. Also note that the `Startup` number is parsed right to left by the handler.

- The two rightmost digits specify the unit number. This determines which DSB expansion serial port will respond to this `devs:mountlist` entry.

  If you are willing to accept the handler's second level of default (`Preferences`), for all serial parameter settings, then all that need be set is the unit number.

- The third digit from the right specifies the default baud rate. A value of 0 causes the handler to use the default baud rate specified by `Preferences`. The other legal values are:

1	110 Baud
2	300 Baud
3	1200 Baud
4	2400 Baud
5	4800 Baud
6	9600 Baud
7	19200 Baud
8	31250 Baud (MIDI)
9	38400 Baud
A	57600 Baud

B	76800 Baud

- The fourth digit from the right specifies the data byte size. This may be the values of 5 through 8 inclusively. If 8 bit data bytes are chosen then the number of stop bits must be 1. If this parameter is 0 then the value set in `Preferences` will be used.
- The fifth digit from the right specifies the type of parity check which will be used. Legal values are 1 through 3 inclusively. They represent no parity, even parity, and odd parity respectively. If this digit is 0 the value set in `Preferences` will be used.
- The sixth digit from the right specifies the number of stop bits which will be used on the serial line. If the number of data bits is chosen to be 8 then the number of stop bits must be 1. Legal values are 1 and 2. If this digit is 0 the value set in `Preferences` will be used.
- The seventh digit from the right specifies the size of the serial receive buffer. These sizes were chosen to correspond to the sizes which can be specified in `Preferences`. If this digit is 0 or is not present, the serial receive buffer size set in `Preferences` will be used. The legal values are:

1	512 Bytes
2	1024 Bytes
3	2048 Bytes
4	4096 Bytes
5	8000 Bytes
6	16000 Bytes

- The eighth digit from the right specifies the type of handshaking which will be used on the serial line. Legal values are 1 through 4 inclusively. If zero is specified then handshaking style will be set to match that specified by `Preferences`. The means of each value are:

1	XON/XOFF Handshaking
2	CTS/RTS Handshaking
3	Both XON/XOFF and CTS/RTS Handshaking
4	No Handshaking

When specifying the value of `Startup`, leading zeros may be deleted.

4.3.3 Mountlist Entries

An entry in `devs:mountlist` is necessary to associate a device name (such as `Line5:`) with a handler and unit number. As described in the previous section, the `devs:mountlist` entry also specifies various serial line parameter defaults.

The following lines are required for buffered serial lines (similar to `ser:`):

```
<devname>: Handler = L:SERX-Handler
           Stacksize = 2000
           Priority = 5
           GlobVec = -1
#
```

The following lines are required for unbuffered serial lines (similar to `aux:`):

```
<devname>: Handler = L:SERXI-Handler
           Stacksize = 2000
           Priority = 5
           GlobVec = -1
#
```

In both cases, `<devname>` refers to any name you wish to associate with the serial line. **Note: The trailing # must be present!**

If a `Startup` line is absent entirely, the unit referred to will default to unit 0.

For example, to set up both an interactive (unbuffered) and a buffered serial device on unit 3 running at 9600 baud with XON/XOFF handshaking and no parity, one would specify:

```
Line3:     Handler = L:SERX-Handler
           Stacksize = 2000
           Priority = 5
           GlobVec = -1
           Startup = 0x10010603
#

MikesTerm: Handler = L:SERXI-Handler
           Stacksize = 2000
           Priority = 5
           GlobVec = -1
           Startup = 0x10010603
#
```

A reference to `Line3:` would result in communications over the buffered line. Referring to `MikesTerm:` would result in communications over the unbuffered (or interactive) line. References to both devices may be made, although not at the same time. In order to use these devices, the following lines would have to be executed from some CLI (like the `Startup-Sequence`, for example):

```
mount Line3:
mount MikesTerm:
```

4.4 The Serial DisPatcher

Provided on the DSB distribution disk is an exclusive software utility which we call the Serial DisPatcher (SDP). Using the SDP you will be able to use *any* communications software (which uses the Amiga's standard serial device) on any DSB serial port with *no* modifications to the executable program.

Also, using SDP you can reroute a serial printer to *any* DSB serial port with total transparancy to the rest of the system.

SDP accomplishes this by impersonating the Amiga's Exec Level driver, `serial.device`. Commodore's device driver is renamed `cbmser.device` and is opened by the SDP should you actually want to use the Amiga's serial port.

It is not possible for the SDP to decrease throughput or affect system performance in any detrimental way (so don't even think it).

4.4.1 Installing The SDP

The installation of the SDP is accomplished by answering "y" to the question "`Install Serial DisPatcher? (y|n)`" asked during the `Install` program.

Answering in the affirmative will cause the `Install` program to replace your `serial.device` with the SDP. A slightly modified version of Commodore's serial.device will be copied into your `devs:` directory under the name `cbmser.device`. Your original Commodore supplied device driver will be renamed `original.serial.device` should you wish to keep it available as a back up.

The slightly modified version of Commodore's serial device driver contained on the distribution disk is the V1.3 `serial.device`. The only modification is to replace the internally occuring string "serial.device" with the string "cbmser.device".

4.4.2 Using The SDP

After installing the SDP you may find it necessary to reboot your system if you had already been using the Amiga's serial device driver since the last time you rebooted.

After having done so, if needed, each time any program attempts to open `serial.device` the SDP will present a requester asking you if you wish to use the standard Amiga serial port. If you click on the "Yes" gadget your program will operate as if nothing had happened over the standard Amiga serial port.

If you click on the "No" gadget, you will be asked to specify the unit number of the DSB port you wish your program to use. The calling program will never be aware that it is *not* using the standard Amiga serial port.

Should any internal error occur, such as not being able to open the requester due to a memory shortage, the SDP is programmed to default to using the Amiga's standard serial port.

Note that some programs open the serial device several times (perhaps once for reading and once, separately, for writing). In this case, the SDP will ask you where to direct the serial activity each time. Simply answer the the SDP's requester the same way each time.

4.4.3 Not Using The SDP

Should you elect not to use the services of the SDP, you may still use any program written to use `serial.device` on the DSB. This can be accomplished by replacing all internally occurring strings having the value of "serial.device" with the string "siosbx.device".

This can be done using any well featured text editor (like ASDG's CygnusEd Professional) or any binary file zapper like John Hodgson's NewZap.

Note that this will allow you to use only DSB port 0 as the unit number to be opened in encoded in the actual instructions of the program. Changing this (without having the source code to the program) is not straight forward. This is one of the chief advantages of using ASDG's Serial DisPatcher (ie: any DSB port can be used).

When using a binary file capable editor (like CygnusEd Professional) be careful to not alter the size of the file in any way or to make changes to any bytes other than the device name. Doing so will surely cause trouble.

5 Electronic Support

ASDG maintains a heavy commitment for on-line electronic support.

At this time, ASDG provides a formal support mechanism on Compuserve and BIX. These networks are check regularly for support requests. On Compuserve, section 2 of the AmigaVendor conference is reserved for ASDG support. On BIX, join the ASDG conference for on-line support.

ASDG maintains an informal support mechanism on American People Link and UseNet in the logical venues. These networks are checked for support requests when time permits. There is no certainty that support requests on these networks will be serviced.

6 Future Directions

To ensure that our customers have the finest support software available, ASDG will provide updates from time to time. This software will be made available through various electronic networking systems.

7 Other Software on the Distribution Disk

As a service to you, we have placed several public domain and/or shareware communications packages on the DSB distribution disk.

ASDG Incorporated cannot provide support assistance for these programs which are provided as is. Any questions or suggestions should be directed to the persons specifically mentioned with each program, **not to ASDG**.

We would encourage you to send the shareware authors a donation if you find their programs useful.

As the packages included on the DSB disk are subject to change, please double click on the `On This Disk` icon in the root directory of the distribution disk.

8 Service and Repair Information

Service and repair assistance can be obtained from ASDG Incorporated by calling: (608) 273 - 6585.

Always contact ASDG before returning a product for service. Please have the following information available when you call:

1. Product name, serial number, and revision number.
2. Your shipping and billing address
3. Your contact name and telephone number

Ship products back in the same container they came in, if at all possible. If the original container is not available, take the following precautions:

1. Place boards in anti-static bags.
2. Allow room for padding material.
3. Send the product, a description of the problem, and the information from your phone conversation with the ASDG service staff to:

ASDG Incorporated
925 Stewart Street
Madison, WI 53713
Attn: Service Department

Dual Serial Board Manual Errata Sheet

6/13/89

1. The last sentence on page 4 currently reads as follows:

   Any bit position which has a zero in the corresponding position in **io_Actual** will be ignored.

   Change this to:

   Any bit position which has a zero in the corresponding position in **io_Offset** will be ignored.

2. The first example on the top of page 5 should read:

```
IOB->io_Command = 0x10; /* or 16 or define your own mnemonic */
IOB->Offset     = 1;    /* specify bit 0 be changed */
IOB->io_Length  = 0;    /* it will be changed to a zero */
DoIO(IOB);
```

3. The second example on the top of page 5 should read:

```
IOB->io_Command = 0x10; /* or 16 or define your own mnemonic */
IOB->io_Offset  = 3;    /* specify bits 0 & 1 be changed */
IOB->io_Length  = 2;    /* bit 1 set to 1 bit 0 set to 0 */
```

4. In the second paragraph of section 4.2.3 **Known Limitations** the manual states that the Amiga's own serial port will drift away from the specified baud rates above 38,400 baud. This was incorrect.

   We have now determined that the 6MHz Dual Serial Boards can get only as close as 51,200 baud when setting 57,600 baud. The 6MHz board can actually communicate with the Amiga's own serial port at 76,800 since the accuracy of the 6MHz DSB's baud rate clock is dead on again at that rate.

   The newer 8MHz Dual Serial Boards will correctly communicate with the Amiga's own serial port at all baud rates between 110 and 115,200 baud.

   You can determine which DSB you have by looking at the board. Looking at the board, if the large serial controller chip designation ends in 6PSC it is a 6MHz board. If it ends in 8PSC then it is an 8MHz board.

5. On page 10, section 4.4.2, the Serial DisPatcher will timeout after 15 seconds. When it times out it will default to the Amiga's internal serial port.

ASDG Incorporated Statement Of Warranty

Effective November 10, 1988

ASDG Incorporated warrants this hardware product against defects in materials or workmanship for a period of one year from receipt by the end user. If ASDG Incorporated receives notice of such defects during the warranty period, ASDG will either, at its option, repair or replace products which prove to be defective.

Exclusions

The above warranty shall not apply to defects resulting from: improper or inadequate maintenance by the customer; customer-supplied software or interfacing; unauthorized modification or misuse; operation outside the environmental specifications for the product; or improper site preparation and maintenance.

Obtaining Warranty Service

The warranty card below must be filled out completely and returned to ASDG, along with a copy of the original purchase reciept (with purchase date circled), before any warranty service can be obtained. The customer must receive a return authorization from ASDG before the product may be returned for repair.

The shipping expenses for the transporting of the defective product to and from the original owner shall be at the owner's expense unless special provisions have been made with ASDG prior to shipping. The customer is also responsible for all shipping charges, duties, and taxes for products returned to ASDG from another country.

Warranty Limitations

ASDG Incorporated makes no other warranty, either expressed or implied, with respect to this product. ASDG specifically disclaims the implied warranties of merchantability and fitness for a particular purpose. Some states or provinces do not allow limitations on the duration of an applied warranty, so the above limitation or exclusion may not apply to you. However, any implied warranty merchantability or fitness is limited to the one-year duration of this written warranty.

This warranty gives you specific legal rights, and you may also have other rights which vary from state to state, or province to province.

- -

ASDG Incorporated Product Warranty Registration Form

Name: ____________________

Company: ____________________

Address: ____________________

City: ____________ State: ____________ Zip Code: ____________

Phone: ____________ Extension: ____________

Product Name: ____________ Date Purchased: ____________

Serial Number: ____________ Where Purchased: ____________

Product Name: ____________ Date Purchased: ____________

Serial Number: ____________ Where Purchased: ____________

Product Name: ____________ Date Purchased: ____________

Serial Number: ____________ Where Purchased: ____________

Comments: ____________________

____

Return To: **ASDG Incorporated** **925 Stewart Street** **Madison, WI 53713** **(608) 273-6585**